

**Universidade do Minho**

Escola de Engenharia

Licenciatura em Engenharia Informática

## **Unidade Curricular de Aprendizagem e Decisão Inteligentes**

Ano Letivo de 2022/2023

## **Conceção de Modelos de Aprendizagem**

### **Grupo 6**

A94942 Miguel Velho Raposo  
A97588 Joana Isabel Freitas Pereira  
A91775 José Pedro Batista Fonte  
A96326 Bernard Ambrósio Georges

9 de outubro de 2023

# Índice

Lista de Figuras . . . . .	3
<b>1 Introdução</b>	<b>4</b>
<b>2 Tarefa[A] - dataset de Renda</b>	<b>5</b>
2.1 Modelo . . . . .	5
2.2 Estudo dos dados . . . . .	6
2.2.1 Estudo Inicial . . . . .	6
2.3 Preparação dos dados . . . . .	11
2.3.1 Objetivo 1 (Price) . . . . .	11
2.3.2 Objetivo 2 (Room Type) . . . . .	11
2.4 Modelação . . . . .	12
2.4.1 Objetivo 1 (Price) . . . . .	12
2.4.2 Objetivo 2 (Room_Type) . . . . .	16
2.5 Avaliação . . . . .	17
<b>3 Tarefa[B] - dataset de Obesidade</b>	<b>18</b>
3.1 Modelo . . . . .	18
3.2 Estudo dos dados . . . . .	18
3.2.1 Estudo Inicial . . . . .	18
3.3 Preparação dos dados . . . . .	23
3.3.1 Remoções . . . . .	24
3.3.2 Modificações . . . . .	24
3.4 Modelação . . . . .	24
3.4.1 Classificação . . . . .	25
3.4.2 Regressão . . . . .	26
3.4.3 Redes Neurais . . . . .	27
3.5 Avaliação . . . . .	28
<b>4 Conclusão</b>	<b>29</b>

## Lista de Figuras

2.1	Tarefa A - Modelo . . . . .	5
2.2	Tarefa A - Exploração Dados - Bar Char . . . . .	8
2.3	Tarefa A - Exploração Dados - Correlation . . . . .	8
2.4	Tarefa A - Exploração Dados - Line Plot . . . . .	9
2.5	Tarefa A - Exploração Dados - Crosstab . . . . .	9
2.6	Tarefa A - Exploração Dados - Scatter Plot . . . . .	10
2.7	Tarefa A - Redes Neurais - Scatter Plot . . . . .	14
2.8	Tarefa A - Segmentação - Scatter Plot . . . . .	15
2.9	Tarefa A - Segmentação - Box Plot . . . . .	16
3.1	Tarefa B - Modelo . . . . .	18
3.2	Tarefa B - Exploração de Dados - Box Plot . . . . .	20
3.3	Tarefa B - Exploração de Dados - Correlation . . . . .	21
3.4	Tarefa B - Exploração de Dados - Crosstab . . . . .	21
3.5	Tarefa B - Exploração dos Dados - Bar Chart . . . . .	22
3.6	Tarefa B - Exploração de Dados - Scatter Plot . . . . .	22
3.7	Tarefa B - Preparação de Dados - Data Explorer Numeric . . . . .	23
3.8	Tarefa B - Preparação de Dados - Data Explorer Nominal . . . . .	23
3.9	Tarefa B - Classificação - Scatter Plot . . . . .	25
3.10	Tarefa B - Regressão - Scatter-Plot . . . . .	27

# 1 Introdução

O presente relatório descreve o trabalho prático realizado pelo Grupo 6 no âmbito da Unidade Curricular de Aprendizagem e Decisão Inteligentes, lecionada no curso de Licenciatura em Engenharia Informática durante o 2º Semestre do ano letivo 2022/2023.

O objetivo deste projeto consiste em extrair conhecimento de conjuntos de dados por meio da aplicação de modelos de aprendizagem abordados ao longo do semestre. O projeto foi dividido em duas fases distintas: na primeira fase, o grupo trabalhou com um conjunto de dados de escolha própria, disponível publicamente, enquanto na segunda fase, o grupo recebeu um conjunto de dados atribuído pela equipe docente da disciplina.

Para garantir uma análise organizada e completa, o grupo utilizou uma parte do método CRISP-DM (Cross Industry Standard Process for Data Mining). No entanto, as fases de estudo do negócio e de desenvolvimento foram ignoradas, pois não eram relevantes para o trabalho em questão.

Em cada uma das fases, o grupo explorou, explicou, analisou e preparou o conjunto de dados em questão. Além disso, os modelos foram configurados e executados utilizando a ferramenta *KNIME*. Após a execução dos modelos, os resultados foram cuidadosamente analisados e o conhecimento obtido foi extrapolado para obter informações significativas e relevantes.

O relatório a seguir apresentará em detalhe as etapas e os resultados de cada fase do projeto, destacando as conclusões alcançadas e as contribuições proporcionadas pelo uso de técnicas de aprendizagem e decisão inteligentes.

## 2 Tarefa[A] - dataset de Renda

### 2.1 Modelo

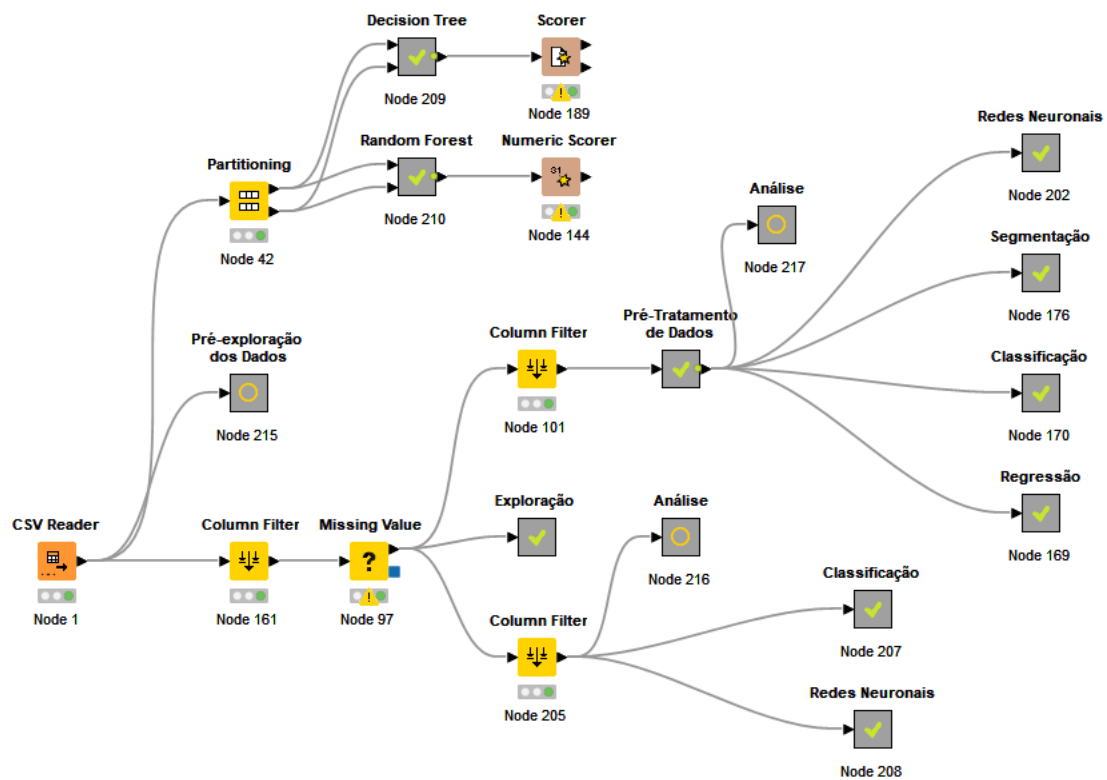


Figura 2.1: Tarefa A - Modelo

## 2.2 Estudo dos dados

### 2.2.1 Estudo Inicial

Na Tarefa[A] o grupo teve de escolher um *dataset* publicamente disponível, para isso, através do Kaggle, o grupo optou por um dataset ligado aos Airbnb em New York City, que contém os seguintes parâmetros:

- **Id** (Int) : identificação do alojamento;
- **Name** (String) : Nome do alojamento;
- **Host\_id** (Int) : Identificação do hospedeiro;
- **Host\_name** (String) : data e tempo do fim da partida;
- **Neighbourhood\_group** (String) : vizinhança onde o alojamento esta localizado;
- **Neighbourhood** (String) : bairro do alojamento;
- **latitude** (Double) : Latitude da localidade do alojamento ;
- **longitude** (Double) : Longitude da localidade do alojamento ;
- **room\_type** (String) : Tipo do quarto disponível para aluguel (Entire home/apt, Private room, Shared room);
- **minimum\_nights** (Int) : mínimo de noites para o aluguel do alojamento;
- **number\_of\_reviews** (Int) : Quantidade de reviews disponiveis;
- **last\_review** (String) : Ultimo review feito;
- **Price** (Int): Preço do alojamento;
- **reviews\_per\_month** (Double) : Reviews feitos por mês;
- **calculated\_host\_listings\_count** (Int) : Quantas listagens que o hospedeiro tem;
- **availability\_365** (int) : Disponibilidade do alojamento alongo do ano em dias(365 dias).

Analisando os diferentes parâmetros disponíveis no *dataset* o grupo tem dois objetivos: um deles é prever os preços dos arrendamento dos locais e outro é prever o tipo de quarto consoante os dados disponibilizados.

Inicialmente o grupo utilizou o nodo *CSV Reader* para carregar o ficheiro AB\_NYC\_2019.csv para o workflow do Knime para de seguida prosseguir para a sua análise.

## Pré-Exploração e Mini-Tratamento de Dados

Tanto o uso do *Statistics* como do *Data Explorer* é importante antes do tratamento de dados pois ajuda a compreender os dados e identificar possíveis erros ou anomalias. Os dois nodos oferecem estatísticas descritivas, como a média, mediana, desvio padrão e até o top e o bottom de cada coluna. Com essas informações, é possível determinar o melhor método de tratamento dos dados para cada variável e identificar quais são as mais relevantes para o modelo final. O tratamento adequado dos dados é fundamental para se obter um modelo final mais preciso e confiável.

No processo de análise inicial dos dados, foram encontrados alguns problemas como valores discrepantes, inconsistências e valores ausentes através de nodos como o *Statistics*, *Data Explore* e *Box Plot*. Em particular, na coluna *last\_review* foram encontrados 10052 valores ausentes, o que representa quase 1/4 dos dados. Para reduzir a perda de dados, optou-se por filtrar essa coluna usando o *Column Filter* e, de seguida, utilizar o nodo *Missing Values* para tratar os valores ausentes de outras colunas. Este nodo foi configurado para remover a linha caso o valor seja uma string e substituir pela média caso seja um número, o que possibilitou a uma melhor exploração dos dados. De seguida foram analisados os nodos utilizados para a exploração dos dados.

### Box Plot

Através do *Box Plot* foi possível visualizar a distribuição de cada variável do *dataset* e identificar valores discrepantes, como *outliers* e padrões em diferentes grupos ou categorias.

## Bar Chart

O *Bar Chart* é usado para visualizar a frequência de cada valor em uma variável categórica. É importante para entender a distribuição de variáveis categóricas. Através da seguinte figura é possível verificar o número de Airbnbs em cada bairro.

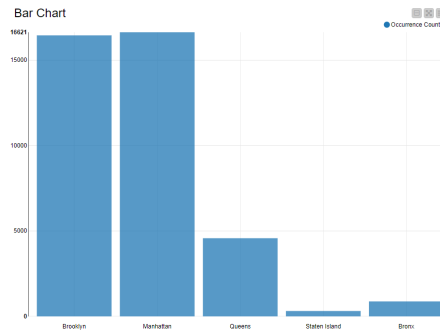


Figura 2.2: Tarefa A - Exploração Dados - Bar Char

## Linear Correlation e Rank Correlation

O nodo *Linear Correlation* foi inicialmente utilizado para calcular a correlação entre as variáveis, porém, como muitas das variáveis continham *missing values*, essa correlação poderia não ser precisa. Por isso, foi decidido utilizar o *Rank Correlation*, que é mais robusto para lidar com esses dados em falta. O *Rank Correlation* avalia as correlações não lineares entre as variáveis, utilizando a ordenação dos valores em vez dos próprios valores. Isso torna o *Rank Correlation* menos sensível a valores discrepantes (*outliers*) e mais adequado para lidar com dados não normalmente distribuídos. Com isso, mediante o nível de correlação entre as colunas é possível selecionar quais devem ser mantidas ou excluídas do modelo, o que permitiu criar modelos mais precisos e confiáveis.

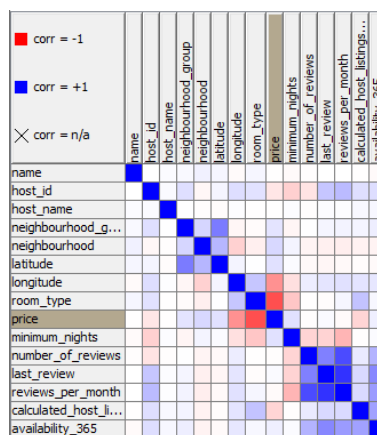


Figura 2.3: Tarefa A - Exploração Dados - Correlation



## GroupBy e Line Plot

O grupo usou o *Group By* para agrupar os dados de acordo com o bairro (*neighbourhood*) e calcular a média dos preços para cada grupo. Em seguida, usou-se o *Line Plot* para visualizar como o preço médio variava em cada bairro. Este nodo é utilizado para criar um gráfico de linha que mostra como uma variável (preço, neste caso) muda em relação a outra (bairro).

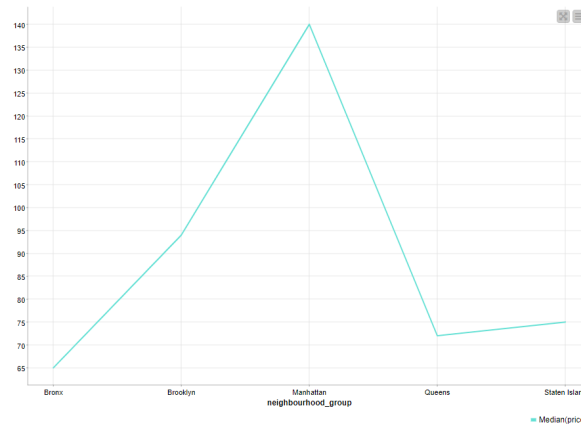


Figura 2.4: Tarefa A - Exploração Dados - Line Plot

## Crosstab

O nodo *Crosstab* é usado para criar uma tabela de contingência entre duas variáveis categóricas. É importante para entender como duas variáveis categóricas estão relacionadas e para identificar possíveis padrões. Assim ajuda a entender melhor os dados e a decidir quais as colunas que devem ser mantidas. O grupo achou pertinente usar a coluna *neighbourhood\_group* e *room\_type* de modo a perceber qual o tipo de quarto mais comum em cada bairro.

Cross Tabulation of room\_type by neighbourhood\_group

Frequency Row Percent	Bronx	Brooklyn	Manhattan	Queens	Staten Island	Total
Entire home/apt	308 1,5157%	8 159 40,1506%	9 962 49,0232%	1 742 8,5724%	150 0,7382%	20 321
Private room	524 2,9682%	7 990 45,2589%	6 303 35,703%	2 678 15,1694%	159 0,9006%	17 654
Shared room	43 5,0827%	290 34,279%	356 42,0804%	152 17,9669%	5 0,591%	846
Total	875	16 439	16 621	4 572	314	38 821

☒ Frequency  
☐ Expected  
☐ Deviation  
☐ Percent  
☒ Row Percent  
☐ Column Percent  
☐ Cell Chi-Square

Max rows: 10  
Max columns: 10

Figura 2.5: Tarefa A - Exploração Dados - Crosstab

## Color Manager e Scatter Plot

O *Color Manager* é usado para codificar as cores usadas no *Scatter Plot*, que é um gráfico bidimensional que mostra a relação entre duas variáveis. É importante para identificar possíveis correlações entre as variáveis. No nosso caso o grupo achou interessante usar as cores para identificar os diferentes tipos de quartos (Entire home/apt - verde; Private room- azul; Shared room - vermelho). De seguida, foi usado o *Scatter Plot* de modo a ser possível verificar a influência do preço no tipo de quarto e nas reviews. Através do gráfico mais à esquerda é possível verificar que preços mais altos coincidem com Entire home/apt e mais reviews com preços mais baixos. Posteriormente foi usado outro *Scatter Plot*, para ver a influência da latitude e longitude no tipo de quarto. Pelo gráfico à direita verifica-se que não parece ter qualquer tipo de relação.

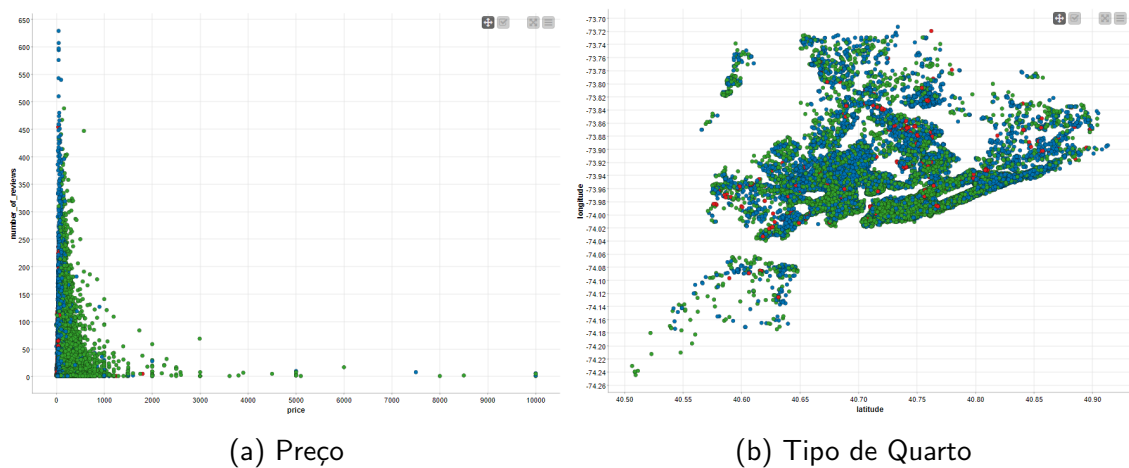


Figura 2.6: Tarefa A - Exploração Dados - Scatter Plot

## 2.3 Preparação dos dados

De modo a corrigir algumas inconsistências encontradas, o grupo realizou uma série de alterações e remoções aos dados de modo a corrigir os problemas nos mesmos para permitir uma extração adequada das informações necessárias para modelo.

De modo a verificar o sucesso dessas alterações e a adequação da preparação dos dados, utilizamos novamente os nodos de *Statistics* e *Data Explorer*.

Dependendo do objetivo foi feita uma preparação dos dados um pouco diferente:

### 2.3.1 Objetivo 1 (Price)

#### Remoções

Inicialmente através do *Column Filter*, como dito anteriormente, o grupo removeu a coluna *last\_review* devido aos imensos missing values.

Através do nodo *Rank Correlation* as colunas que tivessem uma correlação inferior a 0.1 com a coluna *price* foram identificadas como potenciais remoções. De seguida através de outros nodos já explicados na exploração de dados foram escolhidas as seguintes colunas para remoção: *name*, *host\_id*, *host\_name*, *neighbourhood*, *number\_of\_views*, *last\_review*, *reviews\_per\_month*, *availability\_365*

De seguida, através do *Row Filter* foram removidas as linhas que tinham valor 0 na coluna *price*.

Por fim, o grupo achou pertinente o uso do *Numeric Outliers* para remover alguns outliers.

#### Adição

A fim de explorar a influência da distância ao centro da cidade no preço, foi utilizado o nó *Java Snippet* para criar uma nova coluna (*Distância*), que mede a distância de cada Airbnb a um ponto central definido em Nova York (40.7829, -73.9654). Através do *Scatter Plot* foi possível verificar a influência da distância no preço final.

### 2.3.2 Objetivo 2 (Room Type)

#### Remoções

Inicialmente através do *Column Filter*, como dito anteriormente, o grupo removeu a coluna *last\_review* devido aos imensos missing values.

Contudo depois da exploração dos dados, usando o *Rank Correlation*, o *Scatter Plot* e o *Crosstab*, para este tratamento de dados, o grupo achou pertinente remover as seguintes colunas: name, host\_id, host\_name, neighbourhood, number\_of\_views, last\_review, reviews\_per\_month, availability\_365, neighbourhood\_group, latitude e longitude.

## 2.4 Modelação

De modo a realizar uma análise mais rigorosa dos dados, a modelação foi dividida em quatro partes distintas: classificação, regressão, redes neuronais e segmentação.

Esta abordagem permite-nos utilizar as técnicas mais adequadas para cada tipo de análise, de forma a obtermos resultados mais precisos e úteis para o nosso projeto.

### 2.4.1 Objetivo 1 (Price)

#### Regressão

O grupo começou por usar o *Normalizer* para normalizar os dados, com exceção à coluna *price* uma vez que é a variável de destino do modelo. Se a mesma fosse normalizada, isso iria afetar a capacidade do modelo de fazer previsões precisas sobre os preços das propriedades, uma vez que o *Scorer* ia estar a comparar valores desnormalizados do preço com valores normalizados da previsão. Contudo, a ação de normalização é importante para garantir que todas as variáveis tenham o peso ao criar os modelos.

Para a construção do modelo, foram utilizados diferentes learners, como Gradient Boosted Trees Learner (Regression), Random Forest Learner (Regression), Tree Ensemble Learner (Regression), Linear Regression Learner e Polynomial Regression Learner. Esses learners permitiram testar diferentes modelos de regressão e escolher o que melhor se adequava ao problema em questão.

No caso específico mencionado, infelizmente, obtivemos valores de métricas baixos. O modelo que apresentou o melhor desempenho foi o Gradient Boosted Trees Learner, que obteve um  $R^2$  de 0,533 e um MAE de 32,209, por exemplo. Isso significa que esse modelo foi capaz de explicar cerca de 53,3% da variabilidade nos dados e teve um erro médio absoluto de 32,209 na previsão dos preços das propriedades.

Na tentativa de melhorar o modelo, o grupo decidiu usar o *x-Partitioner* e o *X-agregator* em vez do *Partitioning*. No entanto, mesmo efetuando algumas alterações como a mudança da estratégia de amostragem de aleatória para Stratified Sampling e o aumento do número de validações no Cross-validation não se verificou uma melhoria significativa na precisão do modelo.

## Classificação

De modo a usarmos a previsão dos preços para um problema de classificação, o grupo optou por utilizar um *Auto-Binner* para fazer uma divisão dos preços por intervalos. De seguido foi utilizado o *Column Filter* para retirar a coluna price e o *Normalizer* para normalizar os dados, de modo a que todas as variáveis tenham a mesma escala. Isso é importante para garantir que todas as variáveis tenham o mesmo peso ao criar modelos.

Para criar os modelos mencionados acima, os nodos de *Partitioning* foram usados para dividir o conjunto de dados em subconjuntos de treinamento e teste, garantindo que o modelo não fosse treinado e testado no mesmo conjunto de dados, o que poderia levar a uma superestimação do desempenho do modelo. A partição dos dados foi feita em 65-35 e foi usado linear sampling.

De seguida, foram usados vários *learners* (Decision Tree Learner, Random Forest Learner, e o Gradient Boosted Trees Learner) com os seus respetivos *predictors* com o objetivo de usar o nodo *Scorer* para verificar qual seria o melhor a usar consoante os dados disponíveis. Infelizmente todos tiveram uma precisão baixa, a rondar os 50%. O grupo ainda tentou fazer alterações em alguns dos hiperparâmetros mas a diferença foi pouca.

Na tentativa de melhorar o modelo, o grupo decidiu usar o *x-Partitioner* e o *X-agregator* em vez do *Partitioning*. Depois de efetuar algumas alterações como a mudança da estratégia de amostragem de aleatória para Random Sampling e com o número de validações no Cross-validation a 10, entre outras, mas não se verificou uma diferença significativa.

## Redes neuronais

Para o uso deste tipo de modelo foi necessário fazer um pequeno tratamento de dados. Para isso, o grupo utilizou o nodo *Rule Engine* para alterar as colunas, modificando as strings para números inteiros. As seguintes colunas foram alteradas:

- **room\_type:**
  - Private Room → 1
  - Shared Room → 0
  - Entire home/apt → 2
- **neighbourhood\_group:**
  - Brooklyn → 3
  - Manhattan → 2
  - Queens → 1
  - Bronx → 0

De seguida normalizou-se os dados e fez-se a partição dos mesmos.

Usando o *RProp MPL Leraner*, o respetivo predictor e alguns nodos como o *Numeric Scorer* e o *Scatter Plot* foi possível interpretar o desempenho das redes neurais. Os resultados obtidos foram semelhantes aos já verificados anteriormente.

É de salientar que o grupo teve alguma dificuldade nesta parte, uma vez que a normalização da coluna escolhida (price), teve impacto em um resultado negativo no *Numeric Scorer*. Porém o *RProp MPL Leraner* só aceita valores entre 0 e 1, pelo que a coluna tem de ser normalizada. Infelizmente a única solução encontrada foi o uso do *Numeric Scorer* antes da desnormalização dos dados, apesar de parecer incorreto.

Através do seguinte gráfico é possível verificar a influência da distância e do tipo de quarto na previsão do preço. (Entire home/apt - verde, Private room - castanho, Shared room - vermelho)

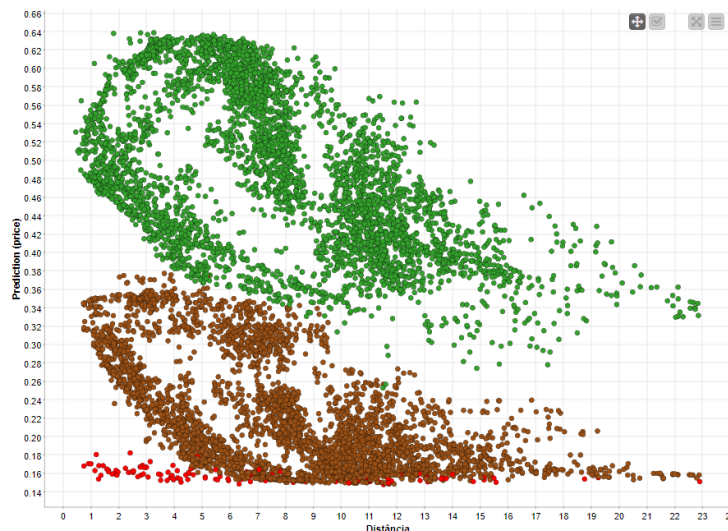


Figura 2.7: Tarefa A - Redes Neurais - Scatter Plot

## Segmentação

O grupo achou que poderia ser interessante usar a segmentação para dividir os dados em clusters que teriam em conta a distância e o preço dos apartamentos. Assim seria possível prever essas categorias através de alguns nodo de classificação.

Para isso, inicialmente usou-se a regra do cotovelo. Porém, dava muitos clusters (12) e optou-se por usar apenas 6 de modo a representar as seguintes categorias: topCenter, mediumCenter, lowCenter, lowSuburb, mediumSuburb, topsuburb.

Depois da desnormalização dos dados, do uso do *K-Means*, do respetivo predictor *Cluster*

*Assigner* e do *Denormalizer* foi efetuada uma análise aos mesmos.

Para essa análise, de modo a perceber melhor como os clusters estavam divididos mediante a distância e o preço, foram usados vários nodos como o *Color Manager*, *Scatter Plot*, *Parallel Coordinates Plot*, *Group By*, *Table View* e *Bar Chart*.

Através do gráfico abaixo é possível fazer a divisão por categorias já mencionada a cima através do nodo *Rule Engine*.

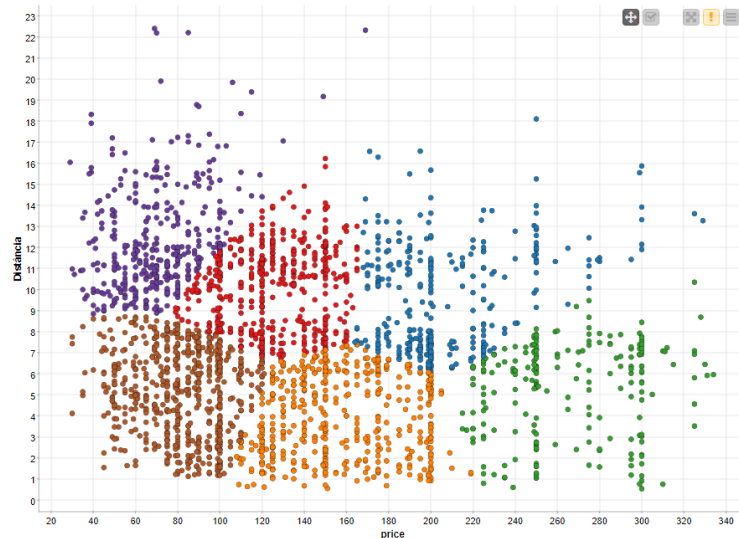


Figura 2.8: Tarefa A - Segmentação - Scatter Plot

Assim as cores atribuídas seriam as seguintes:

- **lowCenter** - castanho;
- **lowSuburb** - roxo;
- **mediumCenter** - laranja;
- **mediumSuburb** - vermelho;
- **topCenter** - verde;
- **topSuburb** - azul

Por fim, foi usado o *Partitioning*, *Decision Tree Learner* e o *Decision Tree Predictor* para avaliar o modelo quantos às categorias criadas. Para isso foram usados nodos como o *Box Plot* e o *Scorer*. Uma vez que a precisão do modelo foi bastante alta (99,2%), o grupo ficou contente com os resultados e considerou que a divisão dos clusters foi bem sucedida.

Pela figura seguinte é possível visualizar que os preços são influenciados pela distância ao centro:

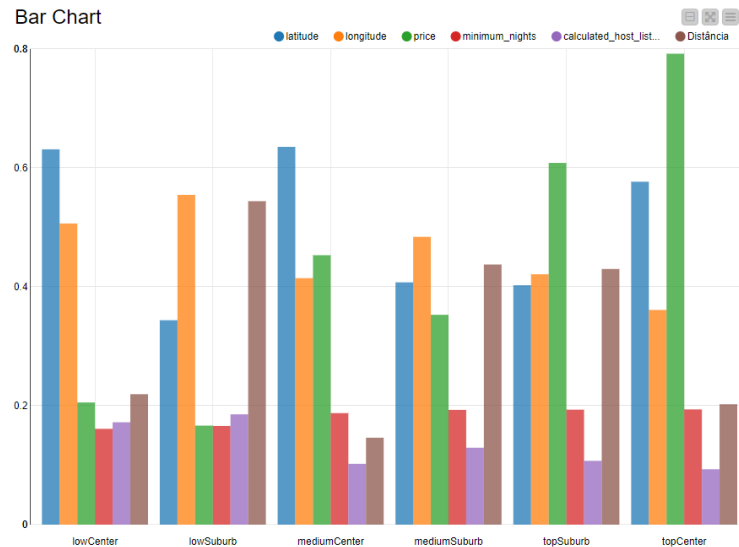


Figura 2.9: Tarefa A - Segmentação - Box Plot

## 2.4.2 Objetivo 2 (Room\_Type)

### Classificação

O grupo começou por usar o *Normalizer* para normalizar os dados, de modo a que todas as variáveis tenham a mesma escala. Isso é importante para garantir que todas as variáveis tenham o mesmo peso ao criar modelos.

De seguida, foi usado um nodo de *Partitioning* para dividir o conjunto de dados em subconjuntos de treinamento e teste, garantindo que o modelo não fosse treinado e testado no mesmo conjunto de dados, o que poderia levar a uma superestimação do desempenho do modelo. A partição dos dados foi feita em 60-40 e foi usado stratified sampling, usando um random seed aleatória.

Posteriormente, foram usados vários *learners* (Decision Tree Learner e o Gradient Boosted Trees Learner) com os seus respetivos *predictors* com o objetivo de usar o nodo *Scorer* para verificar qual seria o melhor a usar consoante os dados disponíveis. O que se destacou mais foi o Gradient Boosted Tree Learner, mas sem uma diferença muito significativa. No primeiro, foi efetuada uma alteração na medida de qualidade, foi alterada do *Gain Ratio* para o *Gini Index* de modo a melhorar a precisão do modelo, já que o segundo é mais adequado para dados com classes desequilibradas. A precisão dos modelos foi por volta dos 80%, um valor bastante aceitável.



## Redes Neurais

Para este caso não foi preciso fazer um tratamento de dados complexo, uma vez todas as colunas, sem contar com a escolhida para a previsão, já se encontravam em double. Foi apenas necessário normalizar os dados.

De seguida, usando o *Partitioning*, o *RProp MPL Learner*, o respetivo predictor e alguns nodos como o *Numeric Scorer* e o *Scatter Plot* foi possível interpretar o desempenho das redes neurais. Os resultados obtidos foram semelhantes aos já verificados anteriormente.

O grupo também tentou o uso do *DL4J Feedforward Learner (Classification)* juntamente com outros nodos, porém verificando a precisão através do nodo *Scorer* foi ligeiramente mais baixa (66% em comparação com os 81% do modelo passado)

## 2.5 Avaliação

É de ainda ressaltar que o *Decision Tree Learner* e o *Random Forest Learner* foram utilizados antes e depois do tratamento de dados para avaliar a eficácia do mesmo e comparar os modelos gerados.

No caso do primeiro, que tem em conta o objetivo2 (room type), as precisões dos modelos não sofreram uma diferença significativa. Isso pode dever-se ao facto de termos eliminado uma coluna importante para a previsão. Mas mantê-la também não seria uma boa opção, pois continha muitos missing values, o que iria levar à perda de muitas linhas, também prejudicial para uma boa previsão. Apesar disso, o grupo considera que o tratamento de dados foi bem-sucedido e que os dados estão mais limpos e bem estruturados, o que leva a modelos mais precisos e confiáveis.

No segundo caso, tendo em conta que objetivo era prever o preço, as precisões dos modelos após o tratamento de dados, apesar de baixas, melhoraram nas várias métricas, como por exemplo de 0,2 para 0,5 no R2, pelo que o tratamento de dados foi bem-sucedido.

Assim pode concluir-se que mediante este *dataset*, e através das colunas que este contém a precisão para prever o tipo de quarto foi bem superior à de prever o preço dos Airbnbs.

## 3 Tarefa[B] - dataset de Obesidade

### 3.1 Modelo

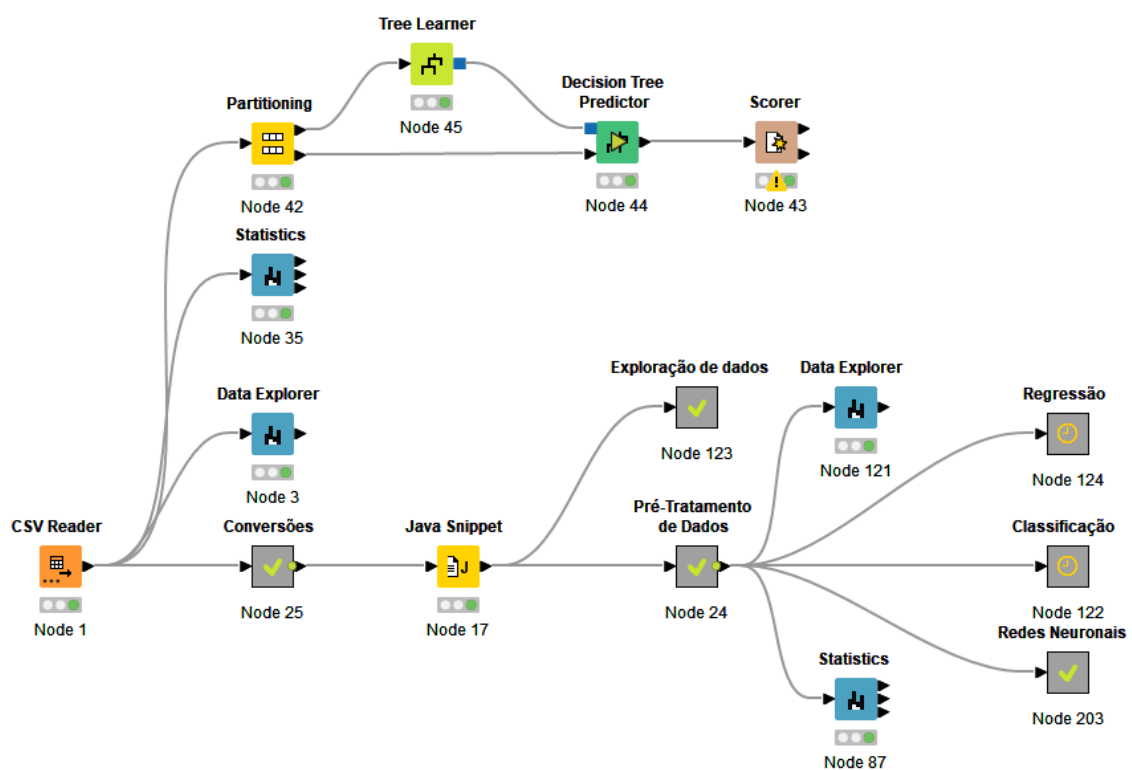


Figura 3.1: Tarefa B - Modelo

### 3.2 Estudo dos dados

#### 3.2.1 Estudo Inicial

Na Tarefa[B] o grupo tem de analisar e preparar um *dataset* que lhe foi atribuído, ligado à obesidade e que contém os seguintes parâmetros:

- **rowID** (int): id do registo;
- **Gender** (string): sexo da pessoa - Male/Female ;
- **Age** (string) : idade em anos;
- **Date\_of\_birth** (string) : data de nascimento (DD/MM/AAAA);
- **Height** (string): altura em metros;
- **Weight** (string) : peso em Kgs;
- **Family\_history\_with\_overweight** (string) : histórico de obesidade na família - yes/no;
- **FAVC** (string): Consumo frequente de alimentos altamente calóricos - yes/no;
- **FCVC** (string): Frequência de consumo de vegetais - sometimes/always/never;
- **NCP** (string): Número de refeições (1-4);
- **CAEC** (string): Consumo de alimentos ente refeições - Sometimes/Frequently/Always/no;
- **SMOKE** (string): Se a pessoa fuma - yes/no
- **CH20** (string): Consumo diário de água (1= menos que 1 litro, 2= 1-2 litros, 3= mais que dois litros);
- **SCC** (string): Monitoramento do consumo de calorias - yes/no
- **FAF** (string): Frequência de atividade física ( 0-nenhuma, 1= 1-2 dias, 2= 2-4 dias, 3= 4-5 dias);
- **TUE** (string): Tempo usado em dispositivos tecnológicos (0= 0-2 horas, 1= 3-5 horas, 2= mais que 5 horas;
- **CALC** (string): Consumo de alcool - Sometimes/no/Frequently/Always;
- **MITRANS** (string): Transporte usado -Public\_Transportation/Automobile/Walking/MotorBike/Bike
- **NObeyesdas** (string): Nível de obesidade - Obesity\_Type\_I/Obesity\_Type\_III,/Obesity\_Type\_II, /Overweight\_Level\_I,/Overweight\_Level\_II,/Normal\_Weight,/Insufficient\_Weight

Analisando os diferentes parâmetros disponíveis no *dataset* o grupo tem como objetivo prever o tipo de obesidade consoante os dados disponibilizados.

Inicialmente o grupo utilizou o nodo *CSV Reader* para carregar o ficheiro *obesidade.csv* para o workflow do *Knime* para de seguida prosseguir para a sua análise.

## Pré-Exploração e Mini-Tratamento de Dados

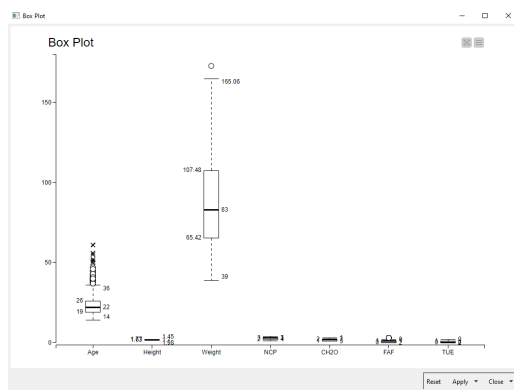
Tanto o uso do *Statistics* como do *Data Explorer* é importante antes do tratamento de dados pois ajuda a compreender os dados e identificar possíveis erros ou anomalias. Os dois nodos oferecem estatísticas descritivas, como a média, mediana, desvio padrão e até o máximo e o mínimo de cada coluna. Com essas informações, é possível determinar o melhor método de tratamento dos dados para cada variável e identificar quais são as mais relevantes para o modelo final. O tratamento adequado dos dados é fundamental para se obter um modelo final mais preciso e confiável.

Através deste nodos foram identificadas possíveis conversões necessárias de modo a ser possível a utilização de outros plots para uma melhor análise dos dados. Utilizando o nodo *String to Number* algumas colunas foram convertidas de *String* para *Double* (Age, Height, Weight, NCP, FAF) para atender ao formato desejado do modelo. Em seguida, utilizando o nodo *Round Double* algumas colunas (Age, NCP, FAF) foram arredondadas para baixo com 0 casas decimais usando o "rounding mode - down". Para as variáveis height e weight, foi usado o "rounding mode - Half\_even" e uma precisão de 2 para arredondar para o número mais próximo, com duas casas decimais.

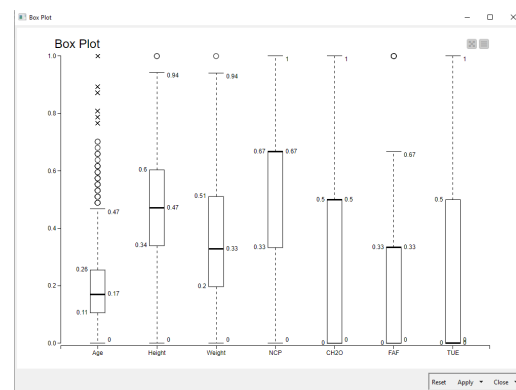
Foram ainda verificados dados danificados na coluna *FCVC* que através do nodo *Java Snippet* foram corrigidos para poder haver uma melhor análise da mesma.

## Box Plot

Através do *Box Plot* foi possível visualizar a distribuição de cada variável do *dataset* e identificar valores discrepantes, como *outliers* e padrões em diferentes grupos ou categorias.



(a) Original



(b) Normalizado

Figura 3.2: Tarefa B - Exploração de Dados - Box Plot

## Linear Correlation e Rank Correlation

O nodo *Linear Correlation* foi inicialmente utilizado para calcular a correlação entre as variáveis, porém, como muitas das variáveis continham *missing values*, essa correlação poderia não ser precisa. Por isso, foi decidido utilizar o *Rank Correlation*, que é mais robusto para lidar com esses dados em falta. O *Rank Correlation* avalia as correlações não lineares entre as variáveis, utilizando a ordenação dos valores em vez dos próprios valores. Isso torna o *Rank Correlation* menos sensível a valores discrepantes (*outliers*) e mais adequado para lidar com dados não normalmente distribuídos. Com isso, mediante o nível de correlação entre as colunas é possível selecionar quais devem ser mantidas ou excluídas do modelo, o que permitiu criar modelos mais precisos e confiáveis.

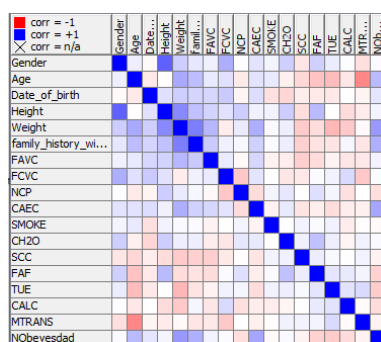


Figura 3.3: Tarefa B - Exploração de Dados - Correlation

## Crosstab

O nodo *Crosstab* é usado para criar uma tabela de contingência entre duas variáveis categóricas. É importante para entender como duas variáveis categóricas estão relacionadas e para identificar possíveis padrões. Assim ajuda a entender melhor os dados e a decidir quais as colunas que devem ser mantidas. Um exemplo do seu uso foi com as colunas *TUE* e *NObesydad* para perceber a relação entre ambas.

Cross Tabulation of TUE by NObesydad

Frequency Row Percent	Insufficient_Weight	Normal_Weight	Obesity_Type_I	Obesity_Type_II	Obesity_Type_III	Overweight_Level_I	Overweight_Level_II	Total
0.0	133 9,3993%	129 9,1166%	221 15,6184%	231 16,3251%	312 22,0495%	204 14,417%	185 13,0742%	1 415
1.0	120 20,4429%	122 20,7836%	105 17,8876%	64 10,9029%	12 2,0443%	70 11,925%	94 16,0136%	587
2.0	19 17,4312%	36 33,0275%	25 22,9358%	2 1,8349%		16 14,6789%	11 10,0917%	109
Total	272	287	351	297	324	290	290	2 111

Figura 3.4: Tarefa B - Exploração de Dados - Crosstab

## Bar Chart

O *Bar Chart* é usado para visualizar a frequência de cada valor em uma variável categórica. É importante para entender a distribuição de variáveis categóricas. Através da seguinte figura é possível verificar o número de pessoas em cada nível de obesidade.

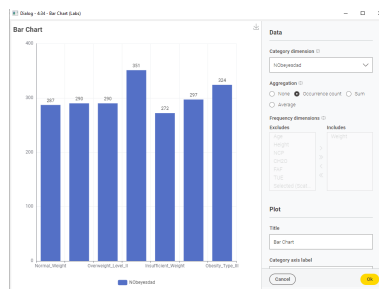


Figura 3.5: Tarefa B - Exploração dos Dados - Bar Chart

## Color Manager e Scatter Plot

O *Color Manager* é usado para codificar as cores usadas no *Scatter Plot*, que é um gráfico bidimensional que mostra a relação entre duas variáveis. É importante para identificar possíveis correlações entre as variáveis. No nosso caso o grupo achou interessante usar as cores para identificar os diferentes tipos de obesidade para de seguida usando o *Scatter Plot* ser possível ver a influência de uma forma mais clara do peso e da altura no nível de obesidade.

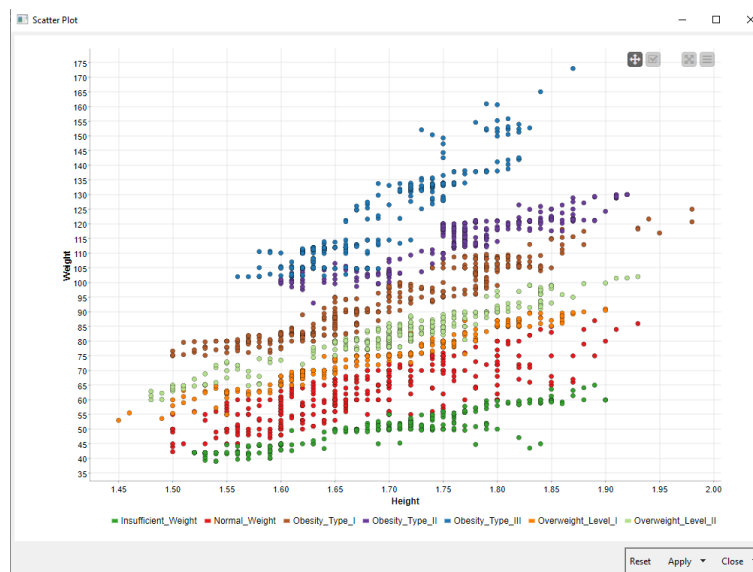


Figura 3.6: Tarefa B - Exploração de Dados - Scatter Plot

### 3.3 Preparação dos dados

De modo a corrigir as algumas inconsistências encontradas, o grupo realizou uma série de alterações e remoções aos dados de modo a corrigir os problemas nos mesmos para permitir uma extração adequada das informações necessárias para modelo.

De modo a verificar o sucesso dessas alterações e a adequação da preparação dos dados, utilizamos novamente os nodos de *Statistics* e *Data Explore*.

Column	Exclude Column	Minimum	Maximum	Mean	Standard Deviation	Variance	Skewness	Kurtosis
Age	<input type="checkbox"/>	14	61	23.973	6.309	39.799	1.562	2.989
Height	<input type="checkbox"/>	1.450	1.980	1.702	0.093	0.009	-0.009	-0.565
Weight	<input type="checkbox"/>	39	173	86.586	26.191	685.977	0.255	-0.700
NCP	<input type="checkbox"/>	1	4	2.523	0.830	0.689	-0.885	-0.457
FAF	<input type="checkbox"/>	0	3	0.735	0.833	0.694	0.899	0.004

Figura 3.7: Tarefa B - Preparação de Dados - Data Explorer Numeric



Column	Exclude Column	No. missings	Unique values	All nominal values	Frequency Bar Chart
family_history_with_overweight	<input type="checkbox"/>	0	2	yes, no	
CAEC	<input type="checkbox"/>	0	4	Sometimes, Frequently, Always, no	
CALC	<input type="checkbox"/>	0	4	Sometimes, no, Frequently, Always	
NObeyesdad	<input type="checkbox"/>	0	7	Obesity_Type_I, Obesity_Type_III, Obesity_Type_II, Overweight_Level_I, Overweight_Level_II, Normal_Weight, Insufficient_Weight	

Figura 3.8: Tarefa B - Preparação de Dados - Data Explorer Nominal

### 3.3.1 Remoções

Através do *Rank Correlation* foi possível identificar as potenciais tabelas a serem removidas, através da fraca correlação entre as variáveis (foram escolhidas as que tinham correlação abaixo de 0.1) e de seguida através do *CrossTab* foi verificada se faria sentido essa remoção. Houve apenas uma exceção, no caso do *TUE*, apesar da correlação ser de 0,2 através do *CrossTab* o grupo achou que faria sentido a sua remoção.

Foi ainda decidido a remoção da coluna *Date\_of\_birth*. Esta remoção deve-se ao facto de já haver uma coluna com a idade do indivíduo e essa ter uma maior correlação com a coluna *NObeyesdas*.

Dito isto as tabelas que foram removidas foram as seguintes: Gender, FCVC, SMOKE, CH20, FACV, SCC, MTRANS, TUE, Date\_of\_birth.

### 3.3.2 Modificações

Em alguns dos campos do *dataset* o grupo teve de recorrer a alterações no mesmo de modo a facilitar a sua análise. As modificações foram realizadas através do nodo *Rule Engine*. As colunas que foram alteradas foram as seguintes:

- **CAEC**: Correção da palavra "sometymes" para "Sometimes";
- **CALC**: Correção da palavra "Frequently" para "Frequently";

## 3.4 Modelação

De modo a realizar uma análise mais rigorosa dos dados, a modelação foi dividida em três partes distintas: classificação, regressão e redes neuronais.

Na parte de classificação, o objetivo é categorizar os dados de acordo com determinados critérios, na parte de regressão o grupo procurou encontrar relações entre variáveis e fazer previsões numéricas e através das redes neuronais o grupo procura explorar a capacidade dos modelos em lidar com problemas complexos e capturar relações não lineares nos dados, contribuindo assim para uma análise mais abrangente e aprimorada do conjunto de dados do nosso projeto.

Esta abordagem permite-nos utilizar as técnicas mais adequadas para cada tipo de análise, de forma a obtermos resultados mais precisos e úteis para o nosso projeto.



### 3.4.1 Classificação

O grupo começou por usar o *Normalizer* para normalizar os dados, de modo a que todas as variáveis tenham a mesma escala. Isso é importante para garantir que todas as variáveis tenham o mesmo peso ao criar modelos.

De seguida, foi usado um nodo de *Partitioning* para dividir o conjunto de dados em subconjuntos de treinamento e teste, garantindo que o modelo não fosse treinado e testado no mesmo conjunto de dados, o que poderia levar a uma superestimação do desempenho do modelo. A partição dos dados foi feita em 75-25 e foi usado stratified sampling, usando um random seed aleatória.

Posteriormente, foram usados vários *learners* (Decision Tree Learner, Random Forest Learner, Tree Ensemble Learner e o Gradient Boosted Trees Learner) com os seus respetivos *predictors* com o objetivo de usar o nodo *Scorer* para verificar qual seria o melhor a usar consoante os dados disponíveis. Os que se destacaram mais foi o o Decision Tree Learner o Gradient Boosted Tree Learner. No primeiro, foi efetuada uma alteração na medida de qualidade, foi alterada do *Gain Ratio* para o *Gini Index* de modo a melhorar a precisão do modelo, já que o segundo é mais adequado para dados com classes desequilibradas. No Gradient Boosted Tree Learner, devido a ser o modelo com melhor performance, para além do *Scorer*, foi usado o nodo *Scatter Plot* para verificar que as previsões estão próximas dos valores reais, pois os pontos no scatter plot estão alinhados próximos a uma linha diagonal, à exceção de alguns.

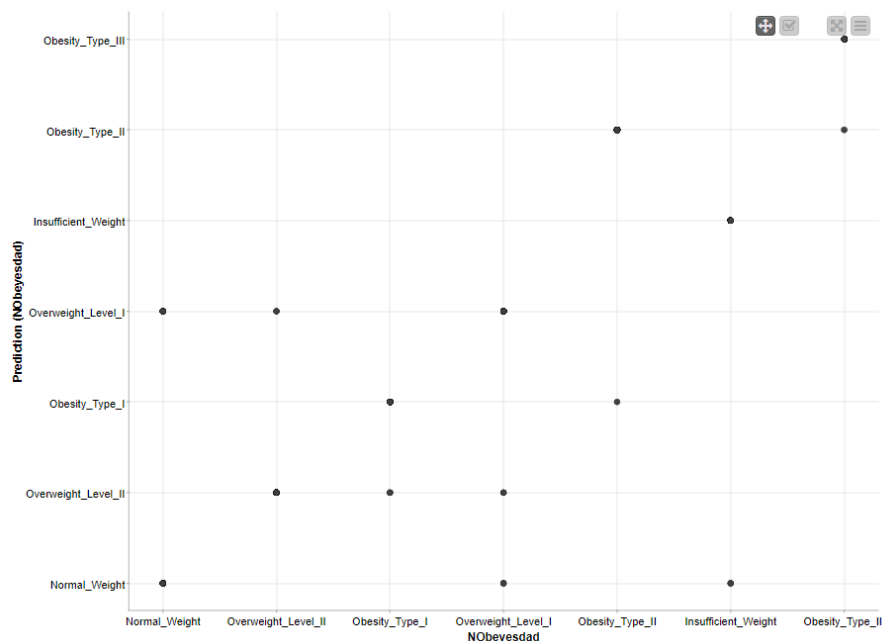


Figura 3.9: Tarefa B - Classificação - Scatter Plot

## X-Partitioner e X-Agregator

Na tentativa de melhorar o modelo, o grupo decidiu usar o *x-Partitioner* e o *X-agregator* em vez do *Partitioning*. Depois de efetuar algumas alterações como a mudança da estratégia de amostragem de aleatória para Stratified Sampling e o aumento do número de validações no Cross-validation verificou-se uma pequena melhoria na precisão, mas nada de significativa.

## Logistic Regression

O algoritmo de regressão logística é uma técnica de aprendizagem supervisionada para prever a probabilidade de uma variável binária. No entanto, como a variável selecionada não era binária, utilizamos o nodo *Rule Engine* para criar uma nova coluna chamada **Obesidade**.

Essa nova coluna foi criada com base na análise da coluna *NObeyesdas*, onde os indivíduos com obesidade foram classificados como "sim" e os restantes como "não". Em seguida, utilizamos o nodo *Column Filter* para remover a coluna *NObeyesdas* e normalizamos os dados.

Por fim, utilizamos o *Logistic Regression Learner* com o predictor correspondente para realizar a análise de regressão logística nos dados normalizados e o *ROC Curve* e o *Scorer* para avaliar a qualidade do modelo obtido. Como era de esperar teve uma precisão bastante elevada, de 99,8%, apenas errando em uma classificação

### 3.4.2 Regressão

O grupo começou por recorrer ao nodo *Java Snippet*, que permitiu a criação de uma nova coluna numérica denominada de índice de massa corporal (IMC). Esse índice foi calculado a partir da altura e do peso de cada indivíduo, transformando assim o objetivo em um problema de regressão, onde a meta é prever o valor do IMC com base nas variáveis.

No entanto, após uma análise exploratória utilizando o *Scatter Plot*, o grupo percebeu que indivíduos com valores de IMC similares eram classificados com diferentes tipos de obesidade.

Nesse sentido, optou-se por criar uma nova tabela contendo a classificação da obesidade com base no IMC, para comparar os resultados. Utilizando o *Scorer*, foi possível verificar que o modelo tinha uma precisão de 80%, indicando que alguns indivíduos estavam mal classificados.

Para a construção do modelo, foram utilizados diferentes learners, como Gradient Boosted Trees Learner (Regression), Random Forest Learner (Regression), Tree Ensemble Learner (Regression), Linear Regression Learner e Polynomial Regression Learner. Esses learners permitiram testar diferentes modelos de regressão e escolher o que melhor se adequava ao problema em questão.

No caso específico mencionado, embora os valores das métricas tenham sido todos bons, o

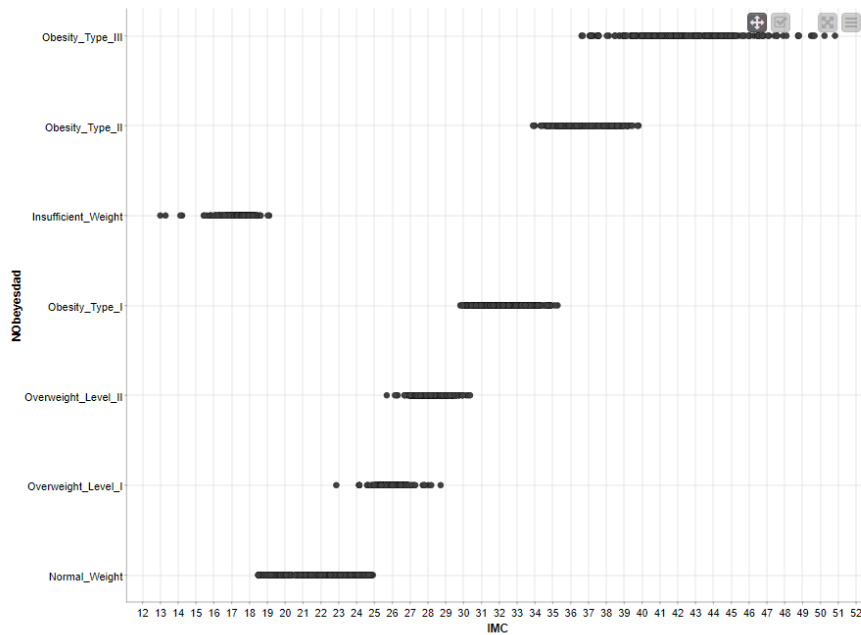


Figura 3.10: Tarefa B - Regressão - Scatter-Plot

modelo que apresentou o melhor desempenho foi o Gradient Boosted Trees Learner, que obteve um  $R^2$  de 0,995 e um MAE de 0,411, entre outros valores bons. Isso indica que esse modelo foi capaz de explicar 99,5% da variação dos dados, e teve um erro médio absoluto de apenas 0,411 unidades, o que é considerado um bom desempenho.

### 3.4.3 Redes Neurais

Para o uso deste tipo de modelo foi necessário fazer um pequeno tratamento de dados. Para isso, o grupo utilizou o nodo *Rule Engine* para alterar as colunas, modificando as strings para números inteiros. As seguintes colunas foram alteradas:

- **family\_history\_with\_overweight:** Alterado de "yes" para 1 e "no" para 0;
- **CAEC e CALC:**
  - Always → 3
  - Sometimes → 1
  - Frequently → 2
  - no → 0

De seguida normalizou-se os dados e fez-se a partição dos mesmos. Usando o *RProp MPL Lerner*, o respetivo predictor e alguns nodos como o *Scorer* e o *Scatter Plot* foi possível interpretar o desempenho das redes neurais na classificação da obesidade. Os resultados obtidos foram bons mas parecidos aos já verificados anteriormente.

## 3.5 Avaliação

É de ainda ressaltar que o *Decision Tree Learner* foi utilizado antes e depois do tratamento de dados para avaliar a eficácia do mesmo e comparar os modelos gerados. Foi identificado uma grande discrepâncias entre estes visto que o *learner* utilizado após o tratamento possui uma precisão bastante maior que o da sua previsão anterior (60%). O que prova que o tratamento de dados foi bem-sucedido e que os dados estão mais limpos e bem estruturados, o que leva a modelos mais precisos e confiáveis.

## 4 Conclusão

A realização deste trabalho proporcionou ao grupo uma oportunidade de aplicar os conhecimentos teóricos adquiridos em um contexto real, enfrentando desafios e tomando decisões sobre a seleção e implementação dos modelos de aprendizagem.

Em algumas partes foram sentidas algumas dificuldades na seleção e implementação de alguns modelos, mas estas foram ultrapassadas e o grupo ficou satisfeito com o trabalho desenvolvido.

Assim, a realização do projeto contribuiu para a consolidação de conhecimentos prévios e para a aquisição de novas habilidades.